

WHAT IS CLAIMED IS:

- 1 1. A computer-implemented method for loading objects in a
2 heterogeneous multiprocessor computer system, said
3 method comprising:
4 identifying a processor to execute a software task,
5 the identification based upon characteristics of the
6 software task and computing resource availability;
7 loading software code corresponding to the identified
8 processor into a shared memory, wherein the shared
9 memory is shared by a plurality of dislike processors
10 that includes the identified processor; and
11 executing the loaded code by the identified processor.
- 1 2. The method as described in claim 1 further comprising:
2 prior to the identifying, compiling a source program
3 into at least two object files, each adapted to be
4 executed on a different processor selected from the
5 plurality of dislike processors, wherein the software
6 code that is loaded and executed is one of the object
7 files.
- 1 3. The method as described in claim 2 further comprising:
2 analyzing the source program for program
3 characteristics; and
4 storing the program characteristics.
- 1 4. The method as described in claim 3 wherein at least
2 one of the program characteristics is selected from

3 the group consisting of data locality, computational
4 intensity, and data parallelism.

1 5. The method as described in claim 3 wherein identifying
2 the processor further comprises:
3 retrieving the program characteristics;
4 retrieving current system characteristics, wherein the
5 current system characteristics includes processor load
6 characteristics for the plurality of dislike
7 processors; and
8 combining the program characteristics and the current
9 system characteristics to determine which of the
10 dislike processors to assign the software task.

1 6. The method as described in claim 5 wherein at least
2 one of the current system characteristics is selected
3 from the group consisting of processor availability
4 for each of the dislike processors, and a data size of
5 data being processed by the software task.

1 7. The method as described in claim 1 further comprising:
2 determining that the identified processor has a
3 scheduler for scheduling tasks for the processor; and
4 scheduling the software code to execute on the
5 identified processor, the scheduling including:
6 writing a software code identifier corresponding
7 to the software code to a run queue corresponding
8 to the identified processor.

1 8. The method as described in claim 1 further comprising:

2 signaling the identified processor;
3 reading, by the identified processor, the software
4 code from the shared memory into a local memory
5 corresponding to the identified processor; and
6 executing the software code by the identified
7 processor.

1 9. The method as described in claim 8 further comprising:
2 writing an instruction block in the shared memory, the
3 instruction block including the address of the loaded
4 software code and the address of an input buffer; and
5 reading the software code and the input buffer from
6 the locations identified in the instruction block to
7 the identified processor 's local memory.

1 10. The method as described in claim 9 further comprising:
2 signaling the identified processor from one of the
3 other processors, the signaling including:
4 writing the address of the instruction block to a
5 mailbox that corresponds to the identified
6 processor; and
7 reading, by the identified processor, the instruction
8 block in response to the signal.

1 11. An information handling system comprising:
2 a plurality of heterogeneous processors;
3 a common memory shared by the plurality of
4 heterogeneous processors;

5 a first processor selected from the plurality of
6 processors that sends a request to a second processor,
7 the second processor also being selected from the
8 plurality of processors;

9 a local memory corresponding to the second processor;

10 a DMA controller associated with the second processor,
11 the DMA controller adapted to transfer data between
12 the common memory and the second processor's local
13 memory; and

14 a loading tool for loading software code to execute on
15 one of the processors, the loading tool including
16 software effective to:

17 identify one of the processors to execute a
18 software task, the identification based upon
19 characteristics of the software task and
20 computing resource availability;

21 loading the software code corresponding to the
22 identified processor into the common memory; and

23 executing the loaded code by the identified
24 processor.

1 12. The information handling system as described in claim
2 11 further comprising software effective to:

3 prior to the identification of one of the processors,
4 a source program compiled into at least two object
5 files, each adapted to be executed on a different
6 processor selected from the plurality of heterogeneous
7 processors, wherein the software code that is loaded
8 and executed is one of the object files.

1 13. The information handling system as described in claim
2 12 further comprising software effective to:
3 analyze the source program for program
4 characteristics; and
5 store the program characteristics.

1 14. The information handling system as described in claim
2 13 wherein at least one of the program characteristics
3 is selected from the group consisting of data
4 locality, computational intensity, and data
5 parallelism.

1 15. The information handling system as described in claim
2 13 wherein identification of the processor further
3 comprises software effective to:
4 retrieve the program characteristics;
5 retrieve current system characteristics, wherein the
6 current system characteristics includes processor load
7 characteristics for the plurality of heterogeneous
8 processors; and
9 combine the program characteristics and the current
10 system characteristics to determine which of the
11 heterogeneous processors to assign the software task.

1 16. The information handling system as described in claim
2 15 wherein at least one of the current system
3 characteristics is selected from the group consisting
4 of processor availability for each of the

5 heterogeneous processors, and a data size of data
6 being processed by the software task.

1 17. The information handling system as described in claim
2 11 further comprising software effective to:

3 determine that the identified processor has a
4 scheduler for scheduling tasks for the processor; and

5 schedule the software code to execute on the
6 identified processor, the schedule including software
7 effective to:

8 write a software code identifier corresponding to
9 the software code to a run queue corresponding to
10 the identified processor.

1 18. The information handling system as described in claim
2 11 further comprising software effective to:

3 signal the identified processor;

4 read, by the identified processor, the software code
5 from the common memory into a local memory
6 corresponding to the identified processor; and

7 execute the software code by the identified processor.

1 19. The information handling system as described in claim
2 18 further comprising software effective to:

3 write an instruction block in the common memory, the
4 instruction block including the address of the loaded
5 software code and the address of an input buffer; and

6 read the software code and the input buffer from the
7 locations identified in the instruction block to the
8 identified processor 's local memory.

1 20. The information handling system as described in claim
2 19 further comprising software effective to:

3 signal the identified processor from one of the other
4 processors, the signal including software effective
5 to:

6 write the address of the instruction block to a
7 mailbox that corresponds to the identified
8 processor; and

9 read, by the identified processor, the instruction
10 block in response to the signal.

1 21. A computer program product stored on a computer
2 operable media for loading objects in a heterogeneous
3 multiprocessor computer system, said computer program
4 product comprising:

5 means for identifying a processor to execute a
6 software task, the identification based upon
7 characteristics of the software task and computing
8 resource availability;

9 means for loading software code corresponding to the
10 identified processor into a shared memory, wherein the
11 shared memory is shared by a plurality of dislike
12 processors that includes the identified processor; and

13 means for executing the loaded code by the identified
14 processor.

1 22. The computer program product as described in claim 21
2 further comprising:

3 prior to the means for identifying, means for
4 compiling a source program into at least two object
5 files, each adapted to be executed on a different
6 processor selected from the plurality of dislike
7 processors, wherein the software code that is loaded
8 and executed is one of the object files.

1 23. The computer program product as described in claim 22
2 further comprising:

3 means for analyzing the source program for program
4 characteristics; and

5 means for storing the program characteristics.

1 24. The computer program product as described in claim 23
2 wherein at least one of the program characteristics is
3 selected from the group consisting of data locality,
4 computational intensity, and data parallelism.

1 25. The computer program product as described in claim 23
2 wherein the means for identifying the processor
3 further comprises:

4 means for retrieving the program characteristics;

5 means for retrieving current system characteristics,
6 wherein the current system characteristics includes
7 processor load characteristics for the plurality of
8 dislike processors; and

9 means for combining the program characteristics and
10 the current system characteristics to determine which
11 of the dislike processors to assign the software task.

1 26. The computer program product as described in claim 25
2 wherein at least one of the current system
3 characteristics is selected from the group consisting
4 of processor availability for each of the dislike
5 processors, and a data size of data being processed by
6 the software task.

1 27. The computer program product as described in claim 21
2 further comprising:

3 means for determining that the identified processor
4 has a scheduler for scheduling tasks for the
5 processor; and

6 means for scheduling the software code to execute on
7 the identified processor, the means for scheduling
8 including:

9 means for writing a software code identifier
10 corresponding to the software code to a run queue
11 corresponding to the identified processor.

1 28. The computer program product as described in claim 21
2 further comprising:

3 means for signaling the identified processor;

4 means for reading, by the identified processor, the
5 software code from the shared memory into a local
6 memory corresponding to the identified processor; and

7 means for executing the software code by the
8 identified processor.

1 29. The computer program product as described in claim 28
2 further comprising:

3 means for writing an instruction block in the shared
4 memory, the instruction block including the address of
5 the loaded software code and the address of an input
6 buffer; and

7 means for reading the software code and the input
8 buffer from the locations identified in the
9 instruction block to the identified processor 's local
10 memory.

1 30. The computer program product as described in claim 29
2 further comprising:

3 means for signaling the identified processor from one
4 of the other processors, the means for signaling
5 including:

6 means for writing the address of the instruction
7 block to a mailbox that corresponds to the
8 identified processor; and

9 means for reading, by the identified processor, the
10 instruction block in response to the signal.